# Backend Authentication: JWT, Tokens, and User Handling - Short Notes

## 1. Mongoose User Schema - Key Concepts

* Schema: A blueprint for documents in MongoDB. Defines what fields a document has.
* Indexing: Makes searching faster (e.g., on `username`, `fullname`).
* Timestamps: Automatically adds `createdAt` and `updatedAt` to each document.
* Reference (ref): Used to link to another model (e.g., `watchHistory` refers to `Video` model).

## 2. Password Handling (Bcrypt)

* Hashing: Encrypting the password into unreadable format for security.
* bcrypt: Library used to hash and verify passwords.
* Salt Rounds: The complexity level of the hash (e.g., 10 rounds).
* Pre-save Hook: Runs before a user is saved, automatically hashes password.
* Password Comparison: Uses `bcrypt.compare()` to check entered password against stored hash.

## 3. JSON Web Tokens (JWT)

* JWT: A compact, self-contained way to securely transmit information between parties.
* Structure: Header + Payload (data) + Signature.
* Signing: Uses a secret key to verify the data is not changed.

## 4. Access Token

* Purpose: Authenticates API requests.
* Short-lived: Typically expires in 10–30 minutes.
* Stored: On client side (like localStorage or memory).
* Used Frequently: Sent with every protected request (usually in headers).

## 5. Refresh Token

* Purpose: Used to get a new Access Token after it expires.
* Long-lived: Can last 7 to 30 days or more.
* Stored Safely: Usually in HTTP-only cookies or DB.
* Issued Alongside: Sent with Access Token during login.
* Used Rarely: Only when Access Token expires.

## 6. Expiry & Secrets

* Set in .env file:
* - ACCESS\_TOKEN\_SECRET=yourSecretKey
* - ACCESS\_TOKEN\_EXPIRY=15m
* - REFRESH\_TOKEN\_SECRET=anotherSecretKey
* - REFRESH\_TOKEN\_EXPIRY=7d
* ExpiresIn defines the validity period of each token.

## 7. Token Lifecycle

* 1. User logs in -> Server generates Access & Refresh tokens.
* 2. Access Token is used to access protected routes.
* 3. When it expires, Refresh Token is sent to get a new Access Token.
* 4. On logout, Refresh Token is deleted or invalidated.

## 8. Why Use Both?

* Access Token: Authenticates requests, short expiry, used frequently.
* Refresh Token: Refreshes session, long expiry, used rarely, needs high security.

## 9. Real-Life Analogy

* Access Token = Movie Ticket (valid for short time).
* Refresh Token = VIP Stamp (lets you re-enter without buying again).
* If ticket expires, show your stamp to get a new one.

## 10. Best Practices

* Never expose refresh tokens to JavaScript (use HTTP-only cookies).
* Rotate refresh tokens regularly.
* Store secrets and expiries in `.env` file.
* Always verify tokens before using.
* Invalidate refresh tokens on logout (set to null in DB).

## Revision Tip

* Keep this sheet handy while building login/auth systems using Node.js, MongoDB, Mongoose, JWT, and bcrypt.